In the Specification:

At page 2, paragraph lines 6-15, please amend line 10 as follows:

Vector architectures have emerged as one approach to reducing the instruction bandwidth required for a computation. With conventional vector architectures, e.g., the Cray-1, a single instruction word specifies a sequence of arithmetic operations, one on each element of a vector of inputs. For example, a vector addition instruction VADD VA, VB, VC causes each element of ~~an~~ a, e.g., sixty-four element vector VA to be added to the corresponding element of a vector VB with the result being placed in the corresponding element of vector VC. Thus, to the extent that the computation being performed can be expressed in terms of vector operations, a vector architecture reduces the required instruction bandwidth by a factor of the vector length (sixty-four in the case of the Cray-1).

At page 6, paragraph lines 13-18, please insert a comma at line 15 as follows:

The above objects are achieved according to a first aspect of the present invention by providing a processor implementing conditional vector operations. In an exemplary conditional vector operation, an input vector containing multiple operands to be used in conditional operations is divided into two or more output vectors based on a condition vector. Each output vector can then be processed at full processor efficiency without cycles wasted due to branch latency.

At page 10, paragraph lines 6-14, please amend the paragraph as follows:

With this foundation in mind, FIG. 1 shows a preferred embodiment of the present invention used in a high speed graphics coprocessor which is described in greater detail in U.S. patent application Ser. No. 09/152,763, now U.S. Patent No. 6,192,384 incorporated herein by reference. Here, a host processor 10 provides data to an image stream processor via a host interface 12. The data from the host processor 10 are stored in a stream register file 14 which is the center of activity in the image stream processor. The host interface 12, a stream memory 16, arithmetic clusters 18, a microcontroller 20 and a network interface 22 all interact by transferring streams of data and instructions to and from the stream register file 14.

At pages 16-17, paragraph lines 12-22 and 1-7 respectively, please insert a comma at page 7, line 3 as follows:

This concept may be implemented in the above-described architecture as follows. First, the microcontroller 20 controls the arithmetic clusters 18 to generate or receive a number of input data values, each having a condition value associated therewith (the condition values are typically computed as the result of comparison operations). Then, a dedicated hardwired circuit performs a parallel prefix scan-+operation as disclosed in, e.g., Cormen et al., Introduction to Algorithms (MIT Electrical Engineering and Computer Science Series), MIT Press, ISBN 0262031418 (incorporated herein by reference) on the condition bits to generate a partial scratchpad index for all data values so that elements having the same condition value, e.g., true or false, are indexed into the same scratchpad area. The preferred embodiment uses a hardwired circuit to perform the scan-+operation because the indices are preferably calculated and the values preferably written into the scratchpad in one cycle, in order to avoid a bottleneck. Thus, a hardware implementation is used. Each partial index is added to the current running index for the vector corresponding to that condition value, and the input data value is stored in the scratchpad register file location pointed to by the absolute index thus obtained via the clusters' inter-cluster communication units 26h. This is done by generating the absolute index so that its least significant three bits denote the destination arithmetic cluster 18 and the remaining bits index into that cluster's scratchpad register file 26g.